HACBPS: A Hierarchical Access Control-Based Proxy Signature

Debasis Giri[1,*], Jiban Dalal[1], P. D. Srivastava[2] and Sanasam Ranbir Singh[3]
[1]Department of Computer science & Engineering
Haldia Institute of Technology, Haldia -721657, India
[2]Department of Mathematics
Indian Institute of Technology, Kharagpur-721302, India
[3]Department of Computer science & Engineering
Indian Institute of Technology, Madras, Chennai-600036, India
Email: debasis_giri@hotmail.com, jiban.dalal@gmail.com, pds@maths.iitkgp.ernet.in and san.ranbir@gmail.com

Abstract— **In this paper, we propose a new security protocol which is styled hierarchical access control-based proxy signature (HACBPS). In hierarchical access control, upper security level users can access some secret information hold by lower security level users, but reverse is not allowed. Whereas in proxy signature, on behalf of the original signer, proxy signer can generate the signature on an arbitrary message. In our protocol, an upper security level user (considered as original signer) can delegate his signing right for signature generation on an arbitrary message to a lower security level user (considered as proxy signer) and the proxy signer can generate proxy signature on behalf of the original signer. In HACBPS, each user in a hierarchy holds two secret keys: one key can be accessed by upper security level users and other one is not accessible to any other user.**

Index Terms— **cryptography, security, access control, proxy signature, Poset**

I. Introduction

[1]In an access control of a hierarchical structure, a user has access some secrets to another if and only if the former is superior of the later. The access control for a hierarchy can be represented by a partially ordered set (Poset). A hierarchy is constructed by dividing users into number disjointed users, say $U_1, U_2, U_3, \ldots, U_n$ which are partially ordered with a binary relation $<=$` . In a hierarchy, $U_i \leq U_j$ means that the security level of U_i is lower than that of. In other words, U_j can access some secret information held by user U_i , while the opposite is not allowed. Figure 1 shows a three level hierarchical structure. The top level user (that is, U_1) poses the highest security and security decreases with increase in level. Hence, users (that is, U_4, U_5, U_6) in bottom level have least security. In 1983, Akl and Taylor first propose hierarchical access-based key assignment scheme [1]. In 1998, Sandhu proposed a tree structural access control scheme [7]. Wu-Wei proposed a scheme [2] which satisfies the indirect access control mechanism. Harn-Lin proposed a scheme [3] which satisfies the direct access control mechanism and is based on the RSA cryptosystem. Giri and Srivastava proposed two schemes: one is access control in tree structural hierarchy [8] and other is poset ordered hierarchy [9] in 2007 and 2008 respectively. Sheng Zhong proposed a scheme [4] which satisfies the indirect access control mechanism. The direct access control schemes achieve smaller storage spaces for storing public information and better dynamics. The access control is motivated by the scenario. A CEO (Chief Executive Officer) of a company can access to some important documents of his General Manager. But General Manager cannot be permitted to access the CEO's documents. In the same manner, General Manager can access the documents of his/her lower level employees, but opposite is strictly prohibited. Whereas, in a proxy signature, proxy signature generation is allowed by a designated person, called a proxy signer, to sign an arbitrary message on behalf of an original signer. An original signer delegates his/her signing capability to a proxy signer (by issuing a proxy key) and then proxy signer signs a message on behalf of the original signer using the proxy key. A verifier can check the validity of that signature and also know the signature which is signed by the proxy signer rather than that by the original signer. More precisely, the original signer sends a specific message with its signature to the proxy signer who then uses this information to construct a proxy signing key. Using the proxy signing key, the proxy signer can generate proxy signatures. From a proxy signature, anyone can verify both the original signer's delegation and the proxy signer's digital signature. The concept of the proxy signature introduces first by Mambo et al. [5, 6]. After that many authors propose many schemes on proxy signatures. In 2008, Giri and Srivastava proposed a proxy signature scheme [10] after removing te weaknesses of Das el al.'s proxy signature scheme [11]. The real life example of proxy signature is as follows: Let CEO of a company wants to ask his

[1]*Corresponding author
Dr. Debasis Giri is at present an Assistant Professor in the Department of Computer Science and Engineering of Haldia Institute of Technology, Haldia-721657, India.

DOI: 01.ijns.01.02.06

General Manager to sign some important documents on his behalf. In this paper, we propose a new security protocol which is combination of hierarchical access control and proxy signature is called access control based proxy signature. But if we simply[2] combine two existing schemes (one is hierarchical access control and other proxy signature) without any change (or new design) then the combined scheme can not be no longer secure, because of fact that secrete key of the lower level security users must be derived (or accessed) by their upper level security users. And so for the proxy signature scheme, if we use the secrete key for the proxy signature generation then the upper level security user can generate the proxy signature because he/she also able to derive (or access) the secrete keys' of its lower level security users. Therefore, the scheme is not proxy protected. But proposed scheme solves that weakness. In our proposed scheme, each user in a hierarchy holds two secret keys: one key can be accessed by upper security level users and other one is not accessible to any other user.

The remainder of this paper is organized as follows. In Section II, we introduce our proposed HACBPS scheme. In Section III, we analyze the security of our proposed scheme. Section IV shows the time complexity required for our scheme. Finally, Section V concludes the paper.

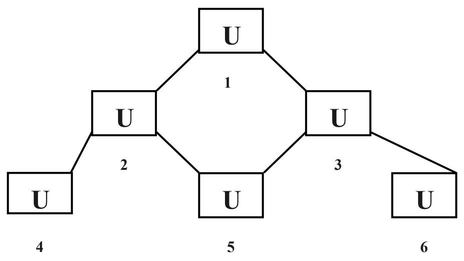

Figure 1: An Example of a Hierarchical Structure

II. Proposed HACBPS Scheme

The HACBPS is the combination of two schemes: first, hierarchical access control scheme; second, a proxy signature scheme. There exists a trusted CA (central authority) in the system that can generate and assign keys for each user in a hierarchy. The scheme consists of eight phases namely, setup, Key assignment by CA, Key derivation by an upper level user, Key generation by a user, Proxy key generation, Proxy key verification, Proxy signature generation, Proxy signature verification.

A. Setup

Let $H(\ )$ be a cryptographic one-way hash function and g a generator of Z_p (where is a large prime of length at least 1024-bit for security consideration).

2

B. Key assignment by CA

Suppose there exists n users in a hierarchy, say $U_1, U_2, \ldots, U_n$. The CA can assign keys for each user in a hierarchy is as follows.

1. CA first choose the root node, that is U_1 (in the rest of the paper, we consider "node" means a user in a hierarchy) and chooses an arbitrary key x_1. CA computes $u_1 = g^{x_1} \bmod p$.
2. Next, CA chooses a node using the technique of breadth first traversal (BFT) of the hierarchical structure. Let U_i be the node chosen by the CA according to BFT.
3. If node U_j is only one direct parent node of U_i, the secret key of U_j is x_i, where $x_i = H(x_j, ID_i)$,

   (1) where ID_i is the identity of the user U_i.
4. If the node U_j has more than one direct parent nodes, say $U_{j1}, U_{j2}, U_{j3}, \ldots, U_{jt}$ where keys of $U_{j1}, U_{j2}, U_{j3}, \ldots, U_{jt}$ are $x_{j1}, x_{j2}, x_{j3}, \ldots, x_{jt}$ respectively, then CA first chooses the secret key x_j for the user U_j. CA then generates the Newton's interpolating polynomial [12] over modulo p containing the points $(H(ID_j \| x_{ji}),\ x_j g^{x_{ji}} \bmod p)$ for $i = 1, 2, \ldots, t$. We denote this polynomial as $P_j(x)$. CA publishes the $P_j(x)$ in a public directory.
5. CA computes $u_i = g^{x_i} \bmod p$.
6. Go to step 2 until all users are not taken consideration in the hierarchy.

Note: CA publishes each $u_i = g^{x_i} \bmod p$ corresponding to the user U_i and sends the secret key x_i to the user U_i (for i=1,2,….) in secure manner.

Example. CA assigns the secret keys for the users corresponding to Figure 1 shown below. CA assigns x_1 for the user U_1. CA then computes $x_2 = H(ID_2 \| x_1)$, $x_3 = H(ID_3 \| x_1)$ for the users U_2, U_3 respectively. CA also computes $x_4 = H(ID_4 \| x_2)$, $x_6 = H(ID_6 \| x_3)$ for the users U_4, U_6 respectively. Finally, CA constructs a Newton's interpolating polynomial containing the points $(H(ID_5 \| x_2),\ x_5 g^{x_2} \bmod p)$ and $(H(ID_5 \| x_3),\ x_5 g^{x_3} \bmod p)$ after choosing the secret key x_5 for the user U_5.

DOI: 01.ijns.01.02.06

C. Key derivation by an upper level user

Suppose $U_i \le U_j$ with a chain $U_i \le U_{k_l} \le \cdots \le U_{k_2} U_{k_1} U_j$. Let C_j want to compute the secret key x_i of the user U_i. U_j first computes the secret key x_{k_1} of the user U_{k_1} using (1), if U_j is the immediate parent of U_{k_1}; otherwise if there are many immediate parents of U_{k_1}, then U_j computes hashed value of $h(ID_{k_l} \| x_j)$ and put the hashed value as x-coordinate in the polynomial $P_{k_1}(x)$. Hence, U_j can get $x_{k_1} g^{x_j} \bmod p$ and then using his secret key x_j, U_j recover the secret key x_{k_1} of the user U_{k_1}. In the similar fashion, using the secret key x_{k_1} (which is computed earlier), U_j computes the secret key x_{k_2} of the uses U_{k_2} and so on until computes the secret key x_i of the user U_i.

D. Key generation by a user

Each user U_j randomly chooses as other secret key and computes the corresponding public key. U_j keeps y_j as a secret key and publishes v_j as public parameter.

E. Proxy key generation

Let U_j be an original signer and U_i a proxy signer. U_j chooses a random number $k(1<k<p-1)$ and computes $K = g^k \bmod p$. He also computes $\sigma = x_j + kx_i + Ky_j m_w \bmod p-1$, (2)
where m_w is a warrant message which consists of the identities of the original as well as proxy signer, expiration date. The original signer U_j delivers the proxy key (σ, m_w, K) to the proxy signer U_i over a public channel.

F. Proxy key verification

The proxy signer checks the condition whether

$$g^{\sigma} = u_j v_j^{Km_w} K^{x_i} \bmod p$$

(3)

is true . If the condition is true, the proxy signer accepts it as a valid proxy; otherwise it is rejected.

G. Proxy signature generation

The proxy signer U_i first chooses z $(1<z<p-1)$ and then computes $w = g^z \bmod p$ and $L_i = K^{x_i} \bmod p$. U_i then computes

$$\sigma' = \sigma + y_i H(m, m_w, v_i, v_j, L_i, w) + z \bmod p-1. \quad (4)$$

He then sends the proxy signature $m, m_w, w, \sigma', L_i, K>$ over a public channel to a verifier.

Note: m_w is a warrant message and w is public information.

H. Proxy signature verification

After receiving $m, m_w, w, \sigma', L_i, K>$ the verifier checks whether the condition

$$g^{\sigma'} = u_j v_j^{Km_w} v_i^{H(m, m_w, v_i, v_j, L_i, w)} L_i w \bmod p \quad (5)$$

holds or not. If it holds good, the verifier accepts it as a valid proxy signature; otherwise it is rejected.

III. Security Analysis

In this section, we describe the security analysis of the proposed scheme.

A. Security for the access control

In our scheme, the key assignment and key derivation by upper level users in a hierarchy are obtained by a cryptographic one-way hash function. If a user U_i has only one direct parent node U_j then the key of U_i will be $x_i = H(x_j, ID_i)$, where x_j is the secret key of U_j. Therefore it is difficult for the user U_i to derive the key x_j of its parent node U_j from $H(x_j, ID_i)$, because of the fact that it is computationally infeasible to invert $H()$. Analogously even a user U_i has many immediate parents nodes, it is also difficult to compute the secret key of any immediate parent node due to the infeasible to invert of a cryptographic one-way hash function.

B. Security analysis for proxy signature

There are six main security properties to be needed for proxy signature such as unforgeability; secret-key's dependency, verifiability, distinguishability, identifiability and unreliability.

In our proposed scheme, each user U_i has a pair of secret keys, x_i, y_i where U_j (with $U_i \le U_j$) can derive x_i using his/her secret key x_j. But U_i cannot compute the secret key y_i from v_i due to discrete logarithm problem (DLP). Further, U_i cannot compute x_j or y_j of the upper level security user U_j.

i) Unforgeability

Suppose an original signer is an adversary ***A***. Now let us check whether ***A*** can forge a proxy signature on an arbitrary message, say m **.**

Suppose ***A*** chooses m', m'_w, L'_i, w', K' and try to find σ'' such that

$$g^{\sigma''} = u_j v_j^{K m'_w} v_i^{H(m', m'_w, v_i, v_j, L'_i, w')} L'_i w' \bmod p \quad (6)$$

holds. Now ***A*** knows all values of the parameters of right hand side (RHS) of (6). Therefore, ***A*** can compute RHS of the equation (6). To compute σ'' such that $\sigma'', m', m'_w, L'_i, w', K'$ satisfies the condition in (6), ***A*** has to solve the DLP (discrete logarithm problem) which is computationally infeasible. Hence, after choosing m', m'_w, L'_i, w', K' it is computationally infeasible to compute σ ¿{} ¿¿¿ such that condition in (6) holds good. Analogously, after choosing any five of *size10σ, m',m'*$_{size8w}$, L'_{size4i}, w, K' , it is also hard to compute the value of the rest such that the condition in (6) holds.

ii) Secret-Key's dependence

In our protocol, original signer derives $\sigma = x_j + kx_i + Ky_j m_w \bmod p-1$ and proxy signerderives $\sigma' = \sigma + y_i H(m, m_w, v_i, v_j, L_i, w) + z \bmod p-1$. From the above equations, it is clear that σ' is computed using σ, y_i with some other public information. σ is derived from x_j, y_j, x_i where (x_j, y_j) is the secrete key pair of the original signer and x_i is one of the secrete key of the proxy signer. So, original signer using his private key can generate a proxy key. It implies that proxy signature key is computed from the secrete key of the original signer. So proxy signature key is secret-key dependent.

iii) Verifiability

From HACBPS scheme, it is clear that proxy signer checks the condition $g^{\sigma} = u_j v_j^{K m_w} K^{x_i} L_i w \bmod p$, where u_j, v_j are the public information corresponding to U_j and K, m_w are the public information. So by these public key, public information and σ , only proxy signer can verify the condition. On the other hands, one can verify the verification condition in (5). Hence the proposed scheme is verifiable.

iv) Distinguisablity

In the verification of the proxy signature the condition, $g^{\sigma'} = u_j v_j^{m_w} v_i^{H(m, m_w, v_i, v_j, L_i, w)} L_i w \bmod p$ is necessary, where σ' is generated by the proxy signer using the equation $\sigma' = \sigma + y_i H(m, m_w, v_i, v_j, L_i, w) + z \bmod p-1$ (where σ is computed by the original signer using the equation $\sigma = x_j + kx_i + Ky_j m_w \bmod p-1$). Hence anyone can verify the proxy signature after receiving $m, m_w, w, \sigma', L_i, K>$. That is to say that a verifier can distinguish a proxy signature rather than the signature generated by the original signer.

v) Identifiability

The verifier can determine the relationship of delegation between an original signer and a proxy signer, because in the verification condition of the proxy signature needs the warrant message m_w which consists of the identity of the original signer as well as proxy signer with expiration date. Hence the verifier can determine that the signature is generated by a proxy signer on behalf of original signer.

vi) Undeniability

In our proxy signature phase, σ' is computed as $\sigma' = \sigma + y_i H(m, m_w, v_i, v_j, L_i, w) + z \bmod p-1$, where y_i is one of the private keys and z a session secret generated by the proxy signer or a lower level user. Involvement of the private key of a proxy signer implies that the proxy signer cannot deny that he has not sign the message. Hence the scheme is undeniable.

IV. Computational Cost

Following are the computation cost needed for the different operation in our scheme.

$t_{\exp}$:	Time taken for a modular exponentiation operation.
t_h :	Time taken for a hashing operation.
t_{mul} :	Time taken for modular multiplication of two numbers.
t_{add} :	Time taken for modular addition of two numbers.

- Computational cost for proxy key generation: $t_{\exp} + 3t_{mul} + 2t_{add}$
- Computational cost for proxy key verification: $2t_{\exp} + 3t_{mul}$
- Computational cost for proxy signature generation: $2t_{\exp} + t_h + t_{mul} + 2t_{add}$
- Computational cost for proxy signature verification: $3t_{\exp} + t_h + 5t_{mul}$

V. Conclusion

In this paper, we have proposed a new security protocol which is called hierarchical access control-based proxy signature. The concept behind the scheme is that anybody in a hierarchy should have two different private keys, where one key can be derived by an upper level user, using key derivation, but other key is only known by the user. The upper level user, after deriving one secrete key from his/her lower level, cannot generate the valid proxy signature because of the fact that other secret key is unknown to the user. We have already discussed the security analysis as well as computational cost of the proposed scheme. Furthermore in our scheme, we can easily adopt dynamicity, that is, some users can be added (or deleted) in (from) a hierarchy.

References

[1] S. G. Akl, and P. D. Taylor, "Cryptographic Solution to a Problem of Access Control in a Hierarchy," ACM Transactions on Computer Systems vol. 1, no. 2, pp. 239-248, 1983.

[2] J. Wu and R. Wei, "An Access Control Scheme for Partially Ordered Set Hierarchy with Provable Security," Selected Areas in Cryptography 2005, LNCS 3897, pp. 221-232, 2005.

[3] L. Harn, and H.-Y. Lin, "A Cryptographic Key Generation Scheme for Multilevel Data Security," Computers and Security, vol. 9, no. 6, pp. 539-546, Oct. 1990.

[4] Sheng Zhong, "A Practical Key Management Scheme for Access Control in a User Hierarchy," Computers & Security, vol. 21, no. 8, pp. 750-759, 2002.

[5] M. Mambo, K. Usuda, and E. Okamoto, "Proxy Signatures Delegation of the Power to Sign Messages," IEICE Trans. Fundamentals, vol. E79-A, no 9, pp. 1339-1353, 1996.

[6] M. Mambo, K. Usuda, and E. Okamoto, "Proxy Signatures for Delegating Signing Operation," in *Proceeding of the 3rd ACM Conference on Computer and Communications Security*, pp. 48-57, 1996.

[7] R. S. Sandhu, "Cryptographic Implementation of a Tree Hierarchy for Access Control", *Information Processing Letters*, no. 27, pp. 95-98, 1988.

[8] D. Giri and P. D. Srivastava, "An Asymmetric Cryptographic Key Assignment Scheme for Access Control in Tree Structural Hierarchies," International Journal of Network Security, Vol.4, No.3, pp.348–354, 2007.

[9] D. Giri and P. D. Srivastava, "A Cryptographic Key Assignment Scheme for Access Control in Poset Ordered Hierarchies with Enhanced Security," International Journal of Network Security, Vol.7, No.2, pp. 223–234, 2008.

[10] D. Giri and P. D. Srivastava, "Cryptanalysis and Improvement of Das et al.'s Proxy Signature Scheme," in the 10th International Conference on Information Technology (ICIT 2007), Rourkela, India, IEEE Computer Society, pp. 151-154, 2007.

[11] M. L. Das, A. Saxena1, and D. B. Phatak, "Proxy Signature Scheme with Effective Revocation using Bilinear Pairings," *International Journal of Network Security*, vol. 4, no. 3, pp. 312-317, 2007.

[12] M. K. Jain, S. R. K. Iyengar, and R. K. Jain, "Numerical Methods for Scientific and Engineering Computation," New Age International Pvt. Ltd. Publisher, 5th Ed., 2007.